

1 Typing Rules

$\Theta; \Delta; \Gamma \vdash e : t$ Typing rules for expressions

$$\begin{array}{c}
\frac{}{\Theta; \Delta; \cdot, x : t \vdash x : t} \text{TY_VAR_LIN} \\
\\
\frac{x : t \in \Delta}{\Theta; \Delta; \cdot \vdash x : t} \text{TY_VAR} \\
\\
\frac{\Theta; \Delta; \Gamma \vdash e : t \quad \Theta; \Delta; \Gamma', x : t \vdash e' : t'}{\Theta; \Delta; \Gamma, \Gamma' \vdash \text{let } x = e \text{ in } e' : t'} \text{TY_LET} \\
\\
\frac{}{\Theta; \Delta; \cdot \vdash () : \text{unit}} \text{TY_UNIT_INTRO} \\
\\
\frac{\Theta; \Delta; \Gamma \vdash e : \text{unit} \quad \Theta; \Delta; \Gamma' \vdash e' : t}{\Theta; \Delta; \Gamma, \Gamma' \vdash \text{let } () = e \text{ in } e' : t} \text{TY_UNIT_ELIM} \\
\\
\frac{}{\Theta; \Delta; \cdot \vdash \text{true} : \text{bool}} \text{TY_BOOL_TRUE} \\
\\
\frac{}{\Theta; \Delta; \cdot \vdash \text{false} : \text{bool}} \text{TY_BOOL_FALSE} \\
\\
\frac{\Theta; \Delta; \Gamma \vdash e : !\text{bool} \quad \Theta; \Delta; \Gamma' \vdash e_1 : t' \quad \Theta; \Delta; \Gamma' \vdash e_2 : t'}{\Theta; \Delta; \Gamma, \Gamma' \vdash \text{if } e \text{ then } e_1 \text{ else } e_2 : t} \text{TY_BOOL_ELIM} \\
\\
\frac{}{\Theta; \Delta; \cdot \vdash k : \text{int}} \text{TY_INT_INTRO} \\
\\
\frac{}{\Theta; \Delta; \cdot \vdash el : \text{elt}} \text{TY_ELT_INTRO} \\
\\
\frac{\Theta; \Delta; \cdot \vdash v : t \quad v \neq l}{\Theta; \Delta; \cdot \vdash \text{Many } v : !t} \text{TY_BANG_INTRO} \\
\\
\frac{\Theta; \Delta; \Gamma \vdash e : !t \quad \Theta; \Delta, x : t; \Gamma' \vdash e' : t'}{\Theta; \Delta; \Gamma, \Gamma' \vdash \text{let Many } x = e \text{ in } e' : t'} \text{TY_BANG_ELIM} \\
\\
\frac{\Theta; \Delta; \Gamma \vdash e : t \quad \Theta; \Delta; \Gamma' \vdash e' : t'}{\Theta; \Delta; \Gamma, \Gamma' \vdash (e, e') : t \otimes t'} \text{TY_PAIR_INTRO} \\
\\
\frac{\Theta; \Delta; \Gamma \vdash e_{12} : t_1 \otimes t_2 \quad \Theta; \Delta; \Gamma', a : t_1, b : t_2 \vdash e : t}{\Theta; \Delta; \Gamma, \Gamma' \vdash \text{let } (a, b) = e_{12} \text{ in } e : t} \text{TY_PAIR_ELIM} \\
\\
\frac{\Theta \vdash t' \text{ Type} \quad \Theta; \Delta; \Gamma, x : t' \vdash e : t}{\Theta; \Delta; \Gamma \vdash \text{fun } x : t' \rightarrow e : t' \multimap t} \text{TY_LAMBDA} \\
\\
\frac{\Theta; \Delta; \Gamma \vdash e : t' \multimap t \quad \Theta; \Delta; \Gamma' \vdash e' : t'}{\Theta; \Delta; \Gamma, \Gamma' \vdash e e' : t} \text{TY_APP}
\end{array}$$

$$\begin{array}{c}
\frac{\Theta, fc; \Delta; \Gamma \vdash e : t}{\Theta; \Delta; \Gamma \vdash \mathbf{fun} \, fc \rightarrow e : \forall fc. t} \quad \text{TY_GEN} \\
\\
\frac{\Theta \vdash f \text{ Cap} \quad \Theta; \Delta; \Gamma \vdash e : \forall fc. t}{\Theta; \Delta; \Gamma \vdash e[f] : t[f/fc]} \quad \text{TY_SPC} \\
\\
\frac{\Theta; \Delta, g : t \multimap t'; \cdot, x : t \vdash e : t'}{\Theta; \Delta; \cdot \vdash \mathbf{fix} \, (g, x : t, e : t') : !(t \multimap t')} \quad \text{TY_FIX}
\end{array}$$

2 Operational Semantics

$\langle \sigma, e \rangle \rightarrow StepsTo$	operational semantics
$\frac{}{\langle \sigma, \text{let } () = () \text{ in } e \rangle \rightarrow \langle \sigma, e \rangle}$	OP_LET_UNIT
$\frac{}{\langle \sigma, \text{let } x = v \text{ in } e \rangle \rightarrow \langle \sigma, e[x/v] \rangle}$	OP_LET_VAR
$\frac{}{\langle \sigma, \text{if } (\text{Many true}) \text{ then } e_1 \text{ else } e_2 \rangle \rightarrow \langle \sigma, e_1 \rangle}$	OP_IF_TRUE
$\frac{}{\langle \sigma, \text{if } (\text{Many false}) \text{ then } e_1 \text{ else } e_2 \rangle \rightarrow \langle \sigma, e_2 \rangle}$	OP_IF_FALSE
$\frac{}{\langle \sigma, \text{let Many } x = \text{Many } v \text{ in } e \rangle \rightarrow \langle \sigma, e[x/v] \rangle}$	OP_LET_MANY
$\frac{}{\langle \sigma, \text{let } (a, b) = (v_1, v_2) \text{ in } e \rangle \rightarrow \langle \sigma, e[a/v_1][b/v_2] \rangle}$	OP_LET_PAIR
$\frac{e_1 = e[g/\text{fun } x : t \rightarrow \text{let Many } g = \text{fix } (g, x : t, e : t') \text{ in } g x]}{\langle \sigma, \text{let Many } g = \text{fix } (g, x : t, e : t') \text{ in } e' \rangle \rightarrow \langle \sigma, e'[g/\text{fun } x : t \rightarrow e_1] \rangle}$	OP_LET_FIX
$\frac{}{\langle \sigma, (\text{fun } fc \rightarrow v)[f] \rangle \rightarrow \langle \sigma, v[fc/f] \rangle}$	OP_FRAC_CAP
$\frac{}{\langle \sigma, (\text{fun } x : t \rightarrow e) v \rangle \rightarrow \langle \sigma, e[x/v] \rangle}$	OP_APP
$\frac{\langle \sigma, e \rangle \rightarrow \langle \sigma', e' \rangle}{\langle \sigma, C[e] \rangle \rightarrow \langle \sigma, C[e'] \rangle}$	OP_CONTEXT
$\frac{\langle \sigma, e \rangle \rightarrow \text{err}}{\langle \sigma, C[e] \rangle \rightarrow \text{err}}$	OP_CONTEXT_ERR
$\frac{0 \leq k_1, k_2}{\langle \sigma, \text{matrix } k_1 k_2 \rangle \rightarrow \langle \sigma \uplus \{l \mapsto_1 M_{k_1, k_2}\}, l \rangle}$	OP_MATRIX
$\frac{}{\langle \sigma \uplus \{l \mapsto_1 m_{k_1, k_2}\}, \text{free } l \rangle \rightarrow \langle \sigma, () \rangle}$	OP_FREE
$\frac{}{\langle \sigma \uplus \{l \mapsto_f m_{k_1, k_2}\}, \text{share } l \rangle \rightarrow \langle \sigma \uplus \{l \mapsto_{\frac{1}{2} \cdot f} m_{k_1, k_2}\} \uplus \{l \mapsto_{\frac{1}{2} \cdot f} m_{k_1, k_2}\}, (l, l) \rangle}$	OP_SHARE
$\frac{f \leq 1}{\langle \sigma \uplus \{l \mapsto_{\frac{1}{2} \cdot f} m_{k_1, k_2}\} \uplus \{l \mapsto_{\frac{1}{2} \cdot f} m_{k_1, k_2}\}, \text{unshare } l \rangle \rightarrow \langle \sigma \uplus \{l \mapsto_f m_{k_1, k_2}\}, l \rangle}$	OP_UNSHARE_EQ
$\frac{l \neq l'}{\langle \sigma \uplus \{l \mapsto_{\frac{1}{2} \cdot f} m_{k_1, k_2}\} \uplus \{l' \mapsto_{\frac{1}{2} \cdot f} m_{k_1, k_2}\}, \text{unshare } l \rangle \rightarrow \text{err}}$	OP_UNSHARE_NEQ
$\frac{\sigma' = \sigma \uplus \{l_1 \mapsto_{fc_1} m_{1k_1, k_2}\} \uplus \{l_2 \mapsto_{fc_2} m_{2k_2, k_3}\}}{\langle \sigma' \uplus \{l_3 \mapsto_1 m_{1k_1, k_3}\}, \text{gemm } l_1 l_2 l_3 \rangle \rightarrow \langle \sigma' \uplus \{l_3 \mapsto_1 (m_1 m_2 + m_3)_{k_1, k_3}\}, ((l_1, l_2), l_3) \rangle}$	OP_GEMM_MATCH
$\frac{k_2 \neq k'_2 \quad \sigma' = \sigma \uplus \{l_1 \mapsto_{fc_1} m_{1k_1, k_2}\} \uplus \{l_2 \mapsto_{fc_2} m_{2k'_2, k_3}\}}{\langle \sigma' \uplus \{l_3 \mapsto_1 m_{1k_1, k_3}\}, \text{gemm } l_1 l_2 l_3 \rangle \rightarrow \text{err}}$	OP_GEMM_MISMATCH

3 Interpretation

$$\mathcal{V}_k[\mathbf{unit}] = \{(\emptyset, *)\}$$

$$\mathcal{V}_k[\mathbf{bool}] = \{(\emptyset, true), (\emptyset, false)\}$$

$$\mathcal{V}_k[\mathbf{int}] = \{(\emptyset, n) \mid 2^{-63} \leq n \leq 2^{63} - 1\}$$

$$\mathcal{V}_k[\mathbf{elt}] = \{(\emptyset, f) \mid f \text{ a IEEE Float64 } \}$$

$$\mathcal{V}_k[f \mathbf{mat}] = \{(\{l \mapsto_{2^{-f}} -\}, l)\}$$

$$\begin{aligned} \mathcal{V}_k[!(t' \multimap t'')] &= \{(\emptyset, \mathbf{Many} \ v) \mid (\emptyset, v) \in \mathcal{V}_k[t' \multimap t'']\} \\ &\cup \{(\emptyset, \mathbf{fix}(g, x : t, e : t')) \mid \forall j < k, (\sigma, v) \in \mathcal{V}_j[t]. \\ &\quad (\sigma, \mathbf{let} \ \mathbf{Many} \ g = \mathbf{fix} \ (g, x : t, e : t') \ \mathbf{in} \ g \ v) \in \mathcal{C}_j[t']\} \end{aligned}$$

$$\mathcal{V}_k[!t] = \{(\emptyset, \mathbf{Many} \ v) \mid \neg(\exists t', t''. t = t' \multimap t'') \wedge (\emptyset, v) \in \mathcal{V}_k[t]\}$$

$$\mathcal{V}_k[\forall fc. t] = \{(\sigma, \mathbf{fun} \ fc \rightarrow v) \mid \forall f. (\sigma, (\mathbf{fun} \ fc \rightarrow v)[f]) \in \mathcal{V}_k[t[fc/f]]\}$$

$$\mathcal{V}_k[t_1 \otimes t_2] = \{(\sigma_1 \star \sigma_2, (v_1, v_2)) \mid (\sigma_1, v_1) \in \mathcal{V}_k[t_1] \wedge (\sigma_2, v_2) \in \mathcal{V}_k[t_2]\}$$

$$\begin{aligned} \mathcal{V}_k[t \multimap t'] &= \{(\sigma, \mathbf{fun} \ x : t \rightarrow e) \mid \forall j < k, (\sigma_v, v) \in \mathcal{V}_j[t]. \sigma \star \sigma_v \text{ defined} \Rightarrow \\ &\quad (\sigma \star \sigma_v, (\mathbf{fun} \ x : t \rightarrow e) \ v) \in \mathcal{C}_j[t']\} \end{aligned}$$

$$\begin{aligned} \mathcal{C}_k[t] &= \{(\sigma_s, e) \mid \forall j \leq k, \sigma_r. \sigma_s \star \sigma_r \text{ defined} \Rightarrow \langle \sigma_s \star \sigma_r, e \rangle \rightarrow^j \mathbf{err} \vee \exists \sigma_f, e'. \\ &\quad \langle \sigma_s \star \sigma_r, e \rangle \rightarrow^j \langle \sigma_f \star \sigma_r, e' \rangle \wedge (e' \text{ is a value} \Rightarrow (\sigma_f \star \sigma_r, e') \in \mathcal{V}_{k-j}[t])\} \end{aligned}$$

$$\mathcal{I}_k[\cdot]\theta = \{\emptyset\}$$

$$\mathcal{I}_k[\Delta, x : t]\theta = \{\delta[x \mapsto v_x] \mid \delta \in \mathcal{I}_k[\Delta]\theta \wedge (\emptyset, v_x) \in \mathcal{V}_k[\theta(t)]\}$$

$$\mathcal{L}_k[\cdot]\theta = \{(\emptyset, \emptyset)\}$$

$$\mathcal{L}_k[\Gamma, x : t]\theta = \{(\sigma \star \sigma_x, \gamma[x \mapsto v_x]) \mid (\sigma, \gamma) \in \mathcal{L}_k[\Gamma]\theta \wedge (\sigma_x, v_x) \in \mathcal{V}_k[\theta(t)]\}$$

$$\begin{aligned} \llbracket \Theta; \Delta; \Gamma \vdash e : t \rrbracket &= \forall \theta, k, \delta, \gamma, \sigma. \text{dom}(\Theta) = \text{dom}(\theta) \wedge (\sigma, \gamma) \in \mathcal{L}_k[\Gamma]\theta \wedge \delta \in \mathcal{I}_k[\Delta]\theta \Rightarrow \\ &\quad (\sigma, \gamma(\delta(e))) \in \mathcal{C}_k[\theta(t)] \end{aligned}$$

4 Soundness Proof

$$\forall \Theta, \Delta, \Gamma, e, t. \Theta; \Delta; \Gamma \vdash e : t \Rightarrow \llbracket \Theta; \Delta; \Gamma \vdash e : t \rrbracket$$

PROOF SKETCH: Induction over the typing judgements.

- ASSUME: 1. Arbitrary $\Theta, \Delta, \Gamma, e, t$ such that $\Theta; \Delta; \Gamma \vdash e : t$.
 2. Arbitrary $\theta, k, \delta, \gamma, \sigma$ such that:
 a. $\text{dom}(\Theta) = \text{dom}(\theta)$
 b. $(\sigma, \gamma) \in \mathcal{L}_k[\Gamma]\theta$
 c. $\delta \in \mathcal{I}_k[\Delta]\theta$.
 3. W.l.o.g., all variables are distinct/ $\text{dom}(\Delta)$ and $\text{dom}(\Gamma)$ are disjoint.
 4. And so that over expressions $\gamma \circ \delta = \delta \circ \gamma$.
 5. By construction, $\text{dom}(\Delta) = \text{dom}(\delta)$ and $\text{dom}(\Gamma) = \text{dom}(\gamma)$.
 6. ??? $\mathcal{V}_k[\theta(t)] \subseteq \mathcal{C}_k[\theta(t)]$.
 7. ??? “Stronger heap”/frame rule: $\langle \sigma, e \rangle \rightarrow^* = \langle \sigma \star \sigma_r, e \rangle \rightarrow^*$.
 8. ??? $\delta(\gamma(v))$ is a value.
 9. ??? $j \leq k \Rightarrow _k[\cdot] \subseteq _j[\cdot]$
 10. ??? $(\sigma', e') \in \mathcal{C}_{k-1}[\cdot] \wedge \langle \sigma, e \rangle \rightarrow \langle \sigma', e' \rangle \Rightarrow (\sigma, e) \in \mathcal{C}_k[\cdot]$

PROVE: $(\sigma, \gamma(\delta(e))) \in \mathcal{C}_k[\theta(t')]$.

ASSUME: Arbitrary $j \leq k$ and σ_r .

SUFFICES: Show whole expression either reduces to **err** or takes j steps.

$\langle 1 \rangle 1$. CASE: **TY_LET**.

PROVE: $(\sigma, \gamma(\delta(\mathbf{let} \ x = e \ \mathbf{in} \ e')))) \in \mathcal{C}_k[\theta(t')]$.

SUFFICES: $(\sigma, \mathbf{let} \ x = \gamma(\delta(e)) \ \mathbf{in} \ \gamma(\delta(e')))) \in \mathcal{C}_k[\theta(t')]$.

$\langle 2 \rangle 1$. By induction,

1. $\llbracket \Theta; \Delta; \Gamma \vdash e : t \rrbracket$
2. $\llbracket \Theta; \Delta; \Gamma', x : t \vdash e' : t' \rrbracket$.

$\langle 2 \rangle 2$. By 2b and induction on Γ' , we know there exist $\sigma_{e'}$, $(\sigma_e, \gamma_e) \in \mathcal{L}_k[\Gamma]$, such that $\sigma = \sigma_e \star \sigma_{e'}$.

$\langle 2 \rangle 3$. So, using them, θ, k, δ , and 3 we have $(\sigma_e, \gamma_e(\delta(e))) \in \mathcal{C}_k[\theta(t)]$.

$\langle 2 \rangle 4$. By 3, $(\sigma_e, \gamma(\delta(e))) \in \mathcal{C}_k[\theta(t)]$.

$\langle 2 \rangle 5$. By definition of $\mathcal{C}_k[\cdot]$ and $\langle 2 \rangle 2$, we instantiate with j and $\sigma_r = \sigma_{e'}$ to conclude that $\langle \sigma, \gamma(\delta(e)) \rangle$ either reduces to **err** or another heap and expression.

$\langle 2 \rangle 6$. CASE: **err**

??? By **OP_CONTEXT_ERR** and 3, the whole expression reduces to **err** in $j \leq k$ steps. Since $j \leq k$ and σ_r (for 7) are arbitrary, $(\sigma, \gamma(\delta(\mathbf{let} \ x = e \ \mathbf{in} \ e')))) \in \mathcal{C}_k[\theta(t')]$.

$\langle 2 \rangle 7$. CASE: j steps to another heap and expression.

By **OP_CONTEXT** and 3, the whole expression does the same.

$\langle 2 \rangle 8$. If it is not a value, we are done. ??? If it is $(\sigma_{ef}, v) \in \mathcal{V}_{k-j}[\theta(t)]$ by 8.

SUFFICES: $(\sigma_{ef} \star \sigma_{e'}, \mathbf{let} \ x = v \ \mathbf{in} \ \gamma(\delta(e')))) \in \mathcal{C}_{k-j}[\theta(t')]$.

SUFFICES: ??? $(\sigma_{ef} \star \sigma_{e'}, \gamma(\delta(e'))[x/v]) \in \mathcal{C}_{k-j-1}[\theta(t')]$ by 10.

$\langle 2 \rangle 9$. DEFINE: $\gamma_{e'}(y) = v$ if $y = x$ and $\gamma(y)$ if $y \in \text{dom}(\Gamma')$.

- ??? Thus, by 9, $(\sigma_{e'}, \gamma_{e'}) \in \mathcal{L}_k[\Gamma', x : t]\theta \subseteq \mathcal{L}_{k-j-1}[\Gamma', x : t]\theta$.
- $\langle 2 \rangle 10$. Instantiate 2 of step $\langle 2 \rangle 1$ with $\theta, k - j - 1, \delta, \gamma_{e'}, \sigma_{e'}$ to conclude $(\sigma_{e'}, \gamma_{e'}(\delta(e')))) \in \mathcal{C}_{k-j-1}[\theta(t')]$.
- $\langle 2 \rangle 11$. By 3, we have $\gamma(\delta(e'))[x/v] = \gamma_{e'}(\delta(e'))$ and by 7 we conclude $(\sigma_{ef} \star \sigma_{e'}, \gamma(\delta(e'))[x/v]) \in \mathcal{C}_{k-j-1}[\theta(t')]$
- $\langle 1 \rangle 2$. CASE: TY_PAIR_ELIM.
 PROVE: $(\sigma, \gamma(\delta(\mathbf{let}(a, b) = e \mathbf{in} e')))) \in \mathcal{C}_k[\theta(t')]$.
 PROOF: Similar to TY_LET but with OP_LET_PAIR
- $\langle 2 \rangle 1$. When $(\sigma_{ef}, v) \in \mathcal{V}_{k-j}[\theta(t_1) \otimes \theta(t_2)]$, we have $v = (v_1, v_2)$.
- $\langle 2 \rangle 2$. SUFFICES: ??? $(\sigma_{e'}, \gamma(\delta(e')))) \in \mathcal{C}_{k-j-1}[\theta(t')]$ by 10.
- $\langle 2 \rangle 3$. DEFINE: $\gamma_{e'}$ to be the restriction of γ to $\text{dom}(\Gamma')$.
 ??? Thus, by 9, $(\sigma_{e'}, \gamma_{e'}[a \mapsto v_1, b \mapsto v_2]) \in \mathcal{L}_k[\Gamma', a : t_1, b : t_2]\theta \subseteq \mathcal{L}_{k-j-1}[\Gamma', a : t_1, b : t_2]\theta$.
- $\langle 2 \rangle 4$. Instantiate $[\Theta; \Delta; \Gamma', a : t_1, b : t_2 \vdash e' : t']$ with $\theta, k - j - 1, \delta, \gamma_{e'}[a \mapsto v_1, b \mapsto v_2], \sigma_{e'}$.
- $\langle 2 \rangle 5$. ??? By 3 $(\sigma_{e'}, \gamma(\delta(e')))) \in \mathcal{C}_{k-j-1}[\theta(t')]$.
- $\langle 1 \rangle 3$. CASE: TY_BANG_ELIM.
 PROVE: $(\sigma, \gamma(\delta(\mathbf{let} \mathbf{Many} x = e \mathbf{in} e')))) \in \mathcal{C}_k[\theta(t)]$.
 PROOF SKETCH: Similar to TY_LET, but with the following key differences.
- $\langle 2 \rangle 1$. When $(\sigma_{ef}, v) \in \mathcal{V}_{k-j}[\theta(!t)]$, since $\mathcal{V}_{k-j}[\theta(!t)] = \mathcal{V}_{k-j}[\theta(t)]$, we have $\sigma_{ef} = \emptyset$ and $v = \mathbf{Many} v'$ for some $(\emptyset, v') \in \mathcal{V}_{k-j}[\theta(t)]$.
- $\langle 2 \rangle 2$. SUFFICES: $(\sigma_{e'}, \mathbf{let} \mathbf{Many} x = \mathbf{Many} v' \mathbf{in} \gamma(\delta(e')))) \in \mathcal{C}_{k-j}[\theta(t)]$.
- $\langle 2 \rangle 3$. SUFFICES: $(\sigma_{e'}, \gamma(\delta(e'))[x/v]) \in \mathcal{C}_{k-j-1}[\theta(t)]$.
- $\langle 2 \rangle 4$. DEFINE: $\gamma_{e'}$ as the restriction of γ to $\text{dom}(\Gamma')$.
- $\langle 2 \rangle 5$. Instantiate $[\Theta; \Delta, x : t, \Gamma' \vdash e' : t']$ with $\theta, k - j - 1, \delta_{e'} = \delta[x \mapsto v'], \gamma_{e'}, \sigma_{e'}$ to conclude $(\sigma_{e'}, \gamma_{e'}(\delta_{e'}(e')))) \in \mathcal{C}_{k-j-1}[\theta(t)]$.
- $\langle 2 \rangle 6$. ??? By 3, $(\sigma_{e'}, \gamma(\delta(e'))[x/v]) \in \mathcal{C}_{k-j-1}[\theta(t)]$.
- $\langle 1 \rangle 4$. CASE: TY_UNIT_ELIM.
 PROVE: $(\sigma, \gamma(\delta(\mathbf{let} () = e \mathbf{in} e')))) \in \mathcal{C}_k[\theta(t)]$.
 PROOF: Similar to TY_LET but with OP_LET_UNIT.
- $\langle 2 \rangle 1$. When $(\sigma_{ef}, v) \in \mathcal{V}_{k-j}[\mathbf{unit}]$, we have $\sigma_{ef} = \emptyset$ and $v = ()$.
- $\langle 2 \rangle 2$. SUFFICES: ??? $(\sigma_{e'}, \gamma(\delta(e')))) \in \mathcal{C}_{k-j-1}[\theta(t')]$ by 10.
- $\langle 2 \rangle 3$. DEFINE: $\gamma_{e'}$ to be the restriction of γ to $\text{dom}(\Gamma')$.
 ??? Thus, by 9, $(\sigma_{e'}, \gamma_{e'}) \in \mathcal{L}_k[\Gamma']\theta \subseteq \mathcal{L}_{k-j-1}[\Gamma']\theta$.
- $\langle 2 \rangle 4$. Instantiate $[\Theta; \Delta; \Gamma' \vdash e' : t']$ with $\theta, k - j - 1, \delta, \gamma_{e'}, \sigma_{e'}$.
- $\langle 2 \rangle 5$. ??? By 3 $(\sigma_{e'}, \gamma(\delta(e')))) \in \mathcal{C}_{k-j-1}[\theta(t')]$.

⟨1⟩5. CASE: TY_BOOL_ELIM.

PROVE: $(\sigma, \gamma(\delta(\text{if } e \text{ then } e_1 \text{ else } e_2))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

PROOF: Similar to TY_UNIT_ELIM but with OP_IF_{\{\text{TRUE}, \text{FALSE}\}} and $\sigma_{ef} = \emptyset$ and $v = \mathbf{Many\ true}$ or $v = \mathbf{Many\ false}$.

⟨1⟩6. CASE: TY_BANG_INTRO.

PROVE: $(\sigma, \gamma(\delta(\mathbf{Many\ } e))) \in \mathcal{C}_k[\![\theta(!t)]\!]$.

SUFFICES: $(\sigma, \mathbf{Many\ } \gamma(\delta(e))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

⟨2⟩1. By assumption of TY_BANG_INTRO, $e = v$ for some value $v \neq l$, $\Gamma = \emptyset$ and so $\llbracket \Theta; \Delta; \cdot \vdash v : t \rrbracket$ by induction.

⟨2⟩2. SUFFICES: $(\emptyset, \mathbf{Many\ } \delta(v)) \in \mathcal{C}_k[\![\theta(t)]\!]$ by 3 and 2b.

⟨2⟩3. Instantiate $\llbracket \Theta; \Delta; \cdot \vdash v : t \rrbracket$ with $\theta, k, \delta, \gamma = \llbracket, \sigma = \emptyset$ to obtain $(\emptyset, \delta(v)) \in \mathcal{C}_k[\![\theta(t)]\!]$.

⟨2⟩4. Instantiate $(\emptyset, \delta(v)) \in \mathcal{C}_k[\![\theta(t)]\!]$ with $j = 0$, and $\sigma_r = \emptyset$, to conclude $(\emptyset, v) \in \mathcal{V}_k[\![\theta(t)]\!]$.

⟨2⟩5. ??? By definition of $\mathcal{V}_k[\![\theta(t)]\!]$, 8 and 6 we have $(\emptyset, \mathbf{Many\ } \delta(v)) \in \mathcal{C}_k[\![\theta(t)]\!]$.

⟨1⟩7. CASE: TY_PAIR_INTRO.

PROVE: $(\sigma, \gamma(\delta((e, e')))) \in \mathcal{C}_k[\![\theta(t \otimes t')]\!]$.

ASSUME: Arbitrary $j \leq k$ and σ_r .

SUFFICES: Show whole expression either reduces to **err** or a heap and expression in j steps.

⟨2⟩1. DEFINE: $(\sigma_1, \gamma_1) \in \mathcal{L}_j[\![\Gamma]\!]$ similar to (σ_e, γ_e) in TY_LET.

⟨2⟩2. By induction,

1. $\llbracket \Theta; \Delta; \Gamma_1 \vdash e_1 : t_1 \rrbracket$
2. $\llbracket \Theta; \Delta; \Gamma_2 \vdash e_2 : t_2 \rrbracket$.

⟨2⟩3. Instantiate the first with $\theta, k, \delta, \gamma_1, \sigma_1$.

⟨2⟩4. Therefore, $(\sigma_1, \gamma_1(\delta(e_1))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

⟨2⟩5. So, $(\sigma_1 \star \sigma_2, \gamma_1(\delta(e_1)))$ either reduces to **err** or a heap and expression in j steps.

⟨2⟩6. CASE: **err**

??? By OP_CONTEXT_ERR and 3, so too does the whole expression. Since $j \leq k$ and σ_r (for 7) are arbitrary, $(\sigma, \gamma(\delta((e, e')))) \in \mathcal{C}_k[\![\theta(t \otimes t')]\!]$.

⟨2⟩7. CASE: j steps to another heap and expression.

By OP_CONTEXT and 3, the whole expression does the same.

⟨2⟩8. If it is not a value, we are done. ??? If it is $(\sigma_{1f}, v_1) \in \mathcal{V}_{k-j}[\![\theta(t_1)]\!]$ by 8.

SUFFICES: ??? By 10, $(\sigma_{1f} \star \sigma_{e_2}, (v_1, e_2)) \in \mathcal{C}_{k-j}[\![\theta(t_1 \otimes t_2)]\!]$.

⟨2⟩9. Instantiate the second IH with $\theta, j, \delta, \gamma_2, \sigma_2$ defined as per usual.

⟨2⟩10. So, $(\sigma_{1f} \star \sigma_2, \gamma_2(\delta(e_2)))$ either reduces to **err** or a heap and expression in j steps.

⟨2⟩11. CASE: **err**

??? By OP_CONTEXT_ERR, 3, so too does the whole expression. Since $j \leq k$ and σ_r (for 7) are arbitrary, $(\sigma_{e_2}, (v_1, e_2)) \in \mathcal{C}_{k-j}[\![\theta(t_1 \otimes t_2)]\!]$.

⟨2⟩12. CASE: j steps to another heap and expression.

By `OP_CONTEXT` and 3, the whole expression does the same.

⟨2⟩13. If it is not a value, we are done. ??? If it is $(\sigma_{2f}, v_2) \in \mathcal{V}_{k-j}[\![\theta(t_2)]\!]$ by 8.
 SUFFICES: ??? By 10, $(\sigma_{1f} \star \sigma_{2f}, (v_1, v_2)) \in \mathcal{C}_{k-2j}[\![\theta(t_1 \otimes t_2)]\!]$.

⟨2⟩14. ??? By 9 and 6, $(\sigma_{1f} \star \sigma_{2f}, (v_1, v_2)) \in \mathcal{V}_{k-j}[\![\cdot]\!] \subseteq \mathcal{V}_{k-2j}[\![\cdot]\!] \subseteq \mathcal{C}_{k-2j}[\![\cdot]\!]$ as needed.

⟨1⟩8. CASE: `TY_LAMBDA`.

PROVE: $(\sigma, \gamma(\delta(\mathbf{fun} x : t \rightarrow e))) \in \mathcal{C}_k[\![\theta(t \multimap t')]\!]$.

SUFFICES: ??? By 6, to show $\dots \in \mathcal{V}_k[\![\theta(t \multimap t')]\!]$.

ASSUME: Arbitrary $j < k$, $(\sigma_v, v) \in \mathcal{V}_j[\![\theta(t)]\!]$ such that $\sigma \star \sigma_v$ is defined.

SUFFICES: $(\sigma \star \sigma_v, \gamma(\delta(\mathbf{fun} x : t \rightarrow e))v) \in \mathcal{C}_j[\![\theta(t')]\!]$.

SUFFICES: $(\sigma \star \sigma_v, \gamma(\delta(e))[x/v]) \in \mathcal{C}_j[\![\theta(t')]\!]$.

⟨2⟩1. By induction, $\llbracket \Theta; \Delta; \Gamma, x : t \vdash e \rrbracket$.

⟨2⟩2. Instantiate it $\theta, j-1, \gamma[x \mapsto v], \sigma_v \star \sigma$.

⟨2⟩3. Hence, $(\sigma_v \star \sigma, \gamma[x \mapsto v](\delta(e))) \in \mathcal{C}_{j-1}[\![\theta(t)]\!]$.

⟨2⟩4. ??? By 3, we are done.

⟨1⟩9. CASE: `TY_APP`.

PROVE: $(\sigma, \gamma(\delta(e e'))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

ASSUME: Arbitrary j and σ_r such that $\sigma \star \sigma_r$ defined.

SUFFICES: Show whole expression either reduces to **err** or a heap and expression in j steps.

⟨2⟩1. By induction,

1. $\llbracket \Theta; \Delta; \Gamma \vdash e : t' \multimap t \rrbracket$
2. $\llbracket \Theta; \Delta; \Gamma' \vdash e' : t' \rrbracket$.

⟨2⟩2. Instantiate the first with $\theta, k, \delta, \gamma_e, \sigma_e$ as per usual definitions,
 to conclude $(\sigma_e, \gamma_e(\delta(e))) \in \mathcal{C}_k[\![\theta(t' \multimap t)]\!]$.

⟨2⟩3. Instantiate *this* with j and $\sigma_{e'}$ to conclude $(\sigma = \sigma_e \star \sigma_{e'}, \gamma(\delta(e e')))$ reduces to **err** or another heap and expression in j steps (using 3).

⟨2⟩4. CASE: **err**

??? By `OP_CONTEXT_ERR`, so too does the whole expression.

Since $j \leq k$ and σ_r (for 7) are arbitrary, $(\sigma, \gamma(\delta(e e'))) \in \mathcal{C}_k[\![\theta(t' \multimap t)]\!]$.

⟨2⟩5. CASE: j steps to another heap and expression.

By `OP_CONTEXT`, the whole expression does the same.

If it is not a value, we are done.

??? If it is $(\sigma_{ef}, \mathbf{fun} x : t \rightarrow e_b) \in \mathcal{V}_{k-j}[\![\theta(t' \multimap t)]\!]$ by 8.

⟨2⟩6. SUFFICES: ??? By 10, to show $(\sigma_{ef} \star \sigma_{e'}, \gamma(\delta((\mathbf{fun} x : t \rightarrow e_b) e')) \in \mathcal{C}_{k-j}[\![\theta(t)]\!]$.

⟨2⟩7. Instantiate the second IH with $\theta, j, \delta, \gamma_{e'}, \sigma_{e'}$ defined as per usual.

⟨2⟩8. So, $(\sigma_{ef} \star \sigma_{e'}, \gamma_{e'}(\delta(e')))$ either reduces to **err** or a heap and expression in j steps.

⟨2⟩9. CASE: **err**

??? By `OP_CONTEXT_ERR` and 3, so too does the whole expression. Since $j \leq k$ and σ_r (for 7) are arbitrary, $(\sigma_{ef} \star \sigma_{e'}, \gamma(\delta((\mathbf{fun} x : t \rightarrow e_b) e')) \in \mathcal{C}_{k-j}[\![\theta(t)]\!]$.

- ⟨2⟩10. CASE: j steps to another heap and expression.
By OP_CONTEXT and 3, the whole expression does the same.
- ⟨2⟩11. If it is not a value, we are done. ??? If it is, by definition of $(\sigma_{ef}, \mathbf{fun} x : t \rightarrow e_b) \in \mathcal{V}_{k-j}[\![\theta(t' \multimap t)]\!]$, we have $(\sigma_{ef} \star \sigma_{e'f}, \gamma(\delta((\mathbf{fun} x : t \rightarrow e_b) v')) \in \mathcal{C}_{k-j}[\![\theta(t)]\!]$.
- ⟨1⟩10. CASE: TY_GEN.
PROVE: $(\sigma, \gamma(\delta(\mathbf{fun} fc \rightarrow e))) \in \mathcal{C}_k[\![\theta(\forall fc. t)]\!]$.
- ⟨1⟩11. CASE: TY_SPC.
PROVE: $(\sigma, \gamma(\delta(e[f]))) \in \mathcal{C}_k[\![\theta(t[fc/f])]\!]$.
- ⟨1⟩12. CASE: TY_FIX.
PROVE: $(\sigma, \gamma(\delta(\mathbf{fix}(g, x : t, e : t')))) \in \mathcal{C}_k[\![\theta(! (t \multimap t'))]\!]$.
SUFFICES: ??? to show $\dots \in \mathcal{V}_k[\![\theta(t) \multimap \theta(t')]\!]$, by 6.
- ⟨2⟩1. ASSUME: Arbitrary $j < k$ and $(\sigma, v) \in \mathcal{V}_j[\![\theta(t)]\!]$.
- ⟨2⟩2. SUFFICES: $(\sigma, \mathbf{let} \mathbf{Many} g = \mathbf{fix}(g, x : t, e : t') \mathbf{in} g v) \in \mathcal{C}_j[\![\theta(t')]\!]$.
- ⟨2⟩3. LET: $e_1 = e[g/\mathbf{fun} x : t \rightarrow \mathbf{let} \mathbf{Many} g = \mathbf{fix}(g, x : t, e : t') \mathbf{in} g x]$.
- ⟨2⟩4. SUFFICES: ??? by 10, $(\sigma, (\mathbf{fun} x : t \rightarrow e_1) v) \in \mathcal{C}_{j-1}[\![\theta(t')]\!]$.
- ⟨2⟩5. SUFFICES: ??? by 10, $(\sigma, e_1[x/v]) \in \mathcal{C}_{j-2}[\![\theta(t')]\!]$.
- ⟨2⟩6. By induction, we have $\llbracket \Theta; \Delta, g : t \multimap t'; x : t \vdash e : t' \rrbracket$.
- ⟨2⟩7. Instantiate this with $\theta, j-2, \delta[g \mapsto \mathbf{fun} x : t \rightarrow e_1], \gamma = [x \mapsto v], \sigma$ (???).
PROVE: $(\sigma, \mathbf{fun} x : t \rightarrow e_1) \in \mathcal{V}_{j-2}[\![\theta(t) \multimap \theta(t')]\!]$.
- ⟨3⟩1. SUFFICES: ??? by 10, $(\sigma', e_1[x/v']) \in \mathcal{C}_{j-2}[\![\theta(t')]\!]$ for arbitrary $(\sigma', v') \in \mathcal{V}_{j-2}[\![\theta(t)]\!]$.
- ⟨3⟩2. We can again use the induction hypothesis $\llbracket \Theta; \Delta, g : t \multimap t'; x : t \vdash e : t' \rrbracket$.
- ⟨3⟩3. But since it's true for $\mathcal{C}_0[\![\cdot]\!]$ (base case), it's true by induction ???
- ⟨2⟩8. Lastly, we show $\delta(\gamma(e)) = e_1[x/v]$, which follows by their definitions,
to conclude $(\sigma, e_1[x/v]) \in \mathcal{C}_{j-2}[\![\theta(t')]\!]$.
- ⟨1⟩13. CASE: TY_VAR_LIN.
PROVE: $(\sigma, \gamma(\delta(x))) \in \mathcal{C}_k[\![\theta(t)]\!]$.
- ⟨2⟩1. $\Gamma = \{x : t\}$ by assumption of TY_VAR_LIN.
- ⟨2⟩2. SUFFICES: $(\sigma, \gamma(x)) \in \mathcal{C}_k[\![\theta(t)]\!]$ by 3.
- ⟨2⟩3. By 2b, there exist $(\sigma_x, v_x) \in \mathcal{V}_k[\![\theta(t)]\!]$, such that $\sigma = \sigma_x$ and $\gamma = [x \mapsto v_x]$.
- ⟨2⟩4. ??? Hence, $(\sigma_x, v_x) \in \mathcal{C}_k[\![\theta(t)]\!]$, by 6.
- ⟨1⟩14. CASE: TY_VAR.
PROVE: $(\sigma, \gamma(\delta(x))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

- $\langle 2 \rangle 1.$ $x : t \in \Delta$ and $\Gamma = \emptyset$ by assumption of `TY_VAR`.
 $\langle 2 \rangle 2.$ SUFFICES: $(\emptyset, \delta(x)) \in \mathcal{C}_k[\![\theta(t)]\!]$ by 3 and 2b.
 $\langle 2 \rangle 3.$ By 2c, there exists v_x such that $(\emptyset, v_x) \in \mathcal{V}_k[\![\theta(t)]\!]$.
 $\langle 2 \rangle 4.$??? Hence, $(\emptyset, v_x) \in \mathcal{C}_k[\![\theta(t)]\!]$, by 6.
- $\langle 1 \rangle 15.$ CASE: `TY_UNIT_INTRO`.
 PROVE: $(\sigma, \gamma(\delta(\ () \))) \in \mathcal{C}_k[\![\theta(\mathbf{unit})]\!]$.
- $\langle 1 \rangle 16.$ CASE: `TY_BOOL_TRUE`, `TY_BOOL_FALSE`, `TY_INT_INTRO`, `TY_ELT_INTRO`.
 Similar to `TY_UNIT_INTRO`.

5 Grammar Definition

m	$::=$		matrix expressions
		M	matrix variables
		$m + m'$	matrix addition
		$m \ m'$	matrix multiplication
		(m)	S
f	$::=$		fractional capability
		fc	variable
		1	whole capability
		$\frac{1}{2} \cdot f$	
t	$::=$		linear type
		unit	unit
		bool	boolean (true/false)
		int	63-bit integers
		elt	array element
		$f \text{ arr}$	arrays
		$f \text{ mat}$	matrices
		$!t$	multiple-use type
		$\forall fc. t$	bind fc in t frac. cap. generalisation
		$t \otimes t'$	pair
		$t \multimap t'$	linear function
		(t)	S parentheses
p	$::=$		primitive
		not	boolean negation
		$(+)$	integer addition
		$(-)$	integer subtraction
		$(*)$	integer multiplication
		$(/)$	integer division
		$(=)$	integer equality
		$(<)$	integer less-than
		$(+.)$	element addition
		$(-.)$	element subtraction
		$(*.)$	element multiplication
		$(/.)$	element division
		$(=.)$	element equality
		$(<.)$	element less-than
		set	array index assignment
		get	array indexing
		share	share array
		unshare	unshare array
		free	free array
		array	Owl: make array
		copy	Owl: copy array
		sin	Owl: map sine over array

	<ul style="list-style-type: none"> hypot asum axpy dot rotmg scal amax setM getM shareM unshareM freeM matrix copyM copyM_to sizeM trnsf gemm symm posv potrs 	<p>Owl: $x_i := \sqrt{x_i^2 + y_i^2}$</p> <p>BLAS: $\sum_i x_i$</p> <p>BLAS: $x := \alpha x + y$</p> <p>BLAS: $x \cdot y$</p> <p>BLAS: see its docs</p> <p>BLAS: $x := \alpha x$</p> <p>BLAS: $\text{argmax } i : x_i$</p> <p>matrix index assignment</p> <p>matrix indexing</p> <p>share matrix</p> <p>unshare matrix</p> <p>free matrix</p> <p>Owl: make matrix</p> <p>Owl: copy matrix</p> <p>Owl: copy matrix onto another</p> <p>dimension of matrix</p> <p>transpose matrix</p> <p>BLAS: $C := \alpha A^{T?} B^{T?} + \beta C$</p> <p>BLAS: $C := \alpha AB + \beta C$</p> <p>BLAS: Cholesky decomp. and solve</p> <p>BLAS: solve with given Cholesky</p>
v	<p>::=</p> <ul style="list-style-type: none"> p x $()$ true false k l el Many v fun $fc \rightarrow v$ $v[f]$ (v, v') fun $x : t \rightarrow e$ bind x in e fix $(g, x : t, e : t')$ bind $g \cup x$ in e (v) S 	<p>values</p> <p>primitives</p> <p>variable</p> <p>unit introduction</p> <p>true</p> <p>false</p> <p>integer</p> <p>heap location</p> <p>array element</p> <p>!-introduction</p> <p>frac. cap. abstraction</p> <p>frac. cap. specialisation</p> <p>pair introduction</p> <p>abstraction</p> <p>fixpoint</p> <p>parentheses</p>
e	<p>::=</p> <ul style="list-style-type: none"> p x let $x = e$ in e' bind x in e' $()$ let $() = e$ in e' true 	<p>expression</p> <p>primitives</p> <p>variable</p> <p>let binding</p> <p>unit introduction</p> <p>unit elimination</p> <p>true</p>

	false $\text{if } e \text{ then } e_1 \text{ else } e_2$ k l el $\text{Many } e$ $\text{let Many } x = e \text{ in } e'$ $\text{fun } fc \rightarrow e$ $e[f]$ (e, e') $\text{let } (a, b) = e \text{ in } e'$ $\text{fun } x : t \rightarrow e$ $e e'$ $\text{fix } (g, x : t, e : t')$ (e)	false if integer heap location array element !-introduction !-elimination $\text{frac. cap. abstraction}$ $\text{frac. cap. specialisation}$ pair introduction pair elimination abstraction application fixpoint parentheses
C	$::=$ $\text{let } x = [-] \text{ in } e$ $\text{let } () = [-] \text{ in } e$ $\text{if } [-] \text{ then } e_1 \text{ else } e_2$ $\text{Many } [-]$ $\text{let Many } x = [-] \text{ in } e$ $\text{fun } fc \rightarrow [-]$ $[-][f]$ $([-], e)$ $(v, [-])$ $\text{let } (a, b) = [-] \text{ in } e$ $[-]e$ $v[-]$	$\text{bind } x \text{ in } e$ $\text{bind } a \cup b \text{ in } e$
		$\text{evaluation contexts}$ let binding unit elimination if !-introduction !-elimination $\text{frac. cap. abstraction}$ $\text{frac. cap. specialisation}$ pair introduction pair introduction pair elimination application application
Θ	$::=$ \cdot Θ, fc	$\text{fractional capability environment}$
Γ	$::=$ \cdot $\Gamma, x : t$ Γ, Γ'	$\text{linear types environment}$
Δ	$::=$ \cdot $\Delta, x : t$	$\text{intuitionistic types environment}$
σ	$::=$ $\{\}$ $\sigma \uplus \{l \mapsto_f m_{k_1, k_2}\}$	heap empty heap $\text{location } l \text{ points to matrix } m$

$StepsTo$	$::=$		result of small step
		$\langle \sigma, e \rangle$	heap and expression
		err	error